Building a Semantic Web Site

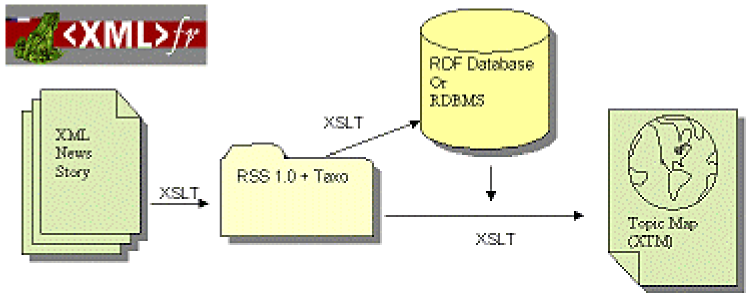
By Eric van der Vlist

Bring Metadata Back to RSS

Even though the Semantic Web may yet seem a remote dream, there are already tools one can use to make a tiny step forward by building "semantic web sites," which can be much easier to navigate than ordinary sites.

In this article, I will discuss how RSS 1.0 and its taxonomy module can be used as a central format to carry metadata collected in a classical news format, such as XMLNews-Story, to RDF or relational databases and XML Topic Maps. Readers should have basic familiarity with RSS and RDF, and a little topic maps knowledge would also help.

Overview



I have built XMLfr (http://xmlfr.org), a French site dedicated to and powered by XML, as a showcase for XML technologies and will use it as a real life example throughout this article. XMLfr is a dynamic site, using XML and XSLT, which stores its pages in the XMLNews-Story format.

The site structure is described by a set of RSS 1.0 channels, and the semantic information encoded in the rich XMLNews-Story inline markup is converted into RSS 1.0 taxonomy markup.

These RSS channels may be consolidated in an RDF database allowing ad hoc semantic queries on the global set of articles. They feed RDBMS tables for online, real-time queries that build a dynamic site index and include navigational information in the XHTML pages sent to the site users.

The RSS channels can be transformed into XTM Topic Maps, to be displayed by Topic Maps visualization systems, and be enriched by the statistics extracted from the database in order to propose topic associations.

About RSS

RSS stands for RDF (or Rich) Site Summary.

Netscape introduced RSS 0.9, one of the first RDF vocabularies, as a general site summary vocabulary in order to syndicate headlines on their "My.Netscape" portal. It was rapidly followed by RSS 0.91 with more syndication features, but leaving out its RDF syntax. Both releases are still widely used as a syndication vocabulary, used by portals such as Userland, Moreover, and Meerkat; but the vocabulary seemed to have reached a dead end by mid-2000.

After the additions of RSS 0.91, the language had lost its focus, many requests for improvement were made without any structure and selection process to advance them, and these requests were pushing in different directions with a risk of loosing still more focus.

More importantly, there was no plan nor method to add metadata.

The RSS 1.0 Working Group (Gabe Beged-Dov, Dan Brickley, Rael Dornfest, Ian Davis, Leigh Dodds, Jonathan Eisenzopf, David Galbraith, R.V. Guha, Ken MacLeod, Eric Miller, Aaron Swartz and myself, Eric van der Vlist) was created with the charter of defining an extensible specification, built on a refocused RDF core vocabulary and a mechanism facilitating the construction of specific modules.

The RSS 1.0 specification (http://purl.org/rss/1.0/) was published in December 2000, together with a Dublin Core module and a set of supporting tools. A taxonomy module is under discussion, and the format used by XMLfr is based on the current Working Draft.

From XMLNews-Story to RSS 1.0

XMLfr's RSS 1.0 channels are generated by an XSLT transformation out of three different sources of information:

- An RSS channel template without any items, and the reference of
- the contents of a directory stored as XML, pointing to
- the XMLNews-Story documents.

The XMLNews-Story element described by the path /nitf/body/body.head contains information that is needed to describe an RSS item, including Dublin Core (DC) elements such as dc:creator, dc:date, dc:description.

The interesting potential of using XMLNews-Story is the possible use of the inline markup to generate more semantic information than is simply specified in the header. XMLfr uses three of these elements that are pertinent to its domain: org, person, and object.title. Extracting these elements allows the generation of dc:object elements in the RSS item's properties to provide a list of keywords for an article. Here's a fragment from an article on XMLfr that shows these elements in use:

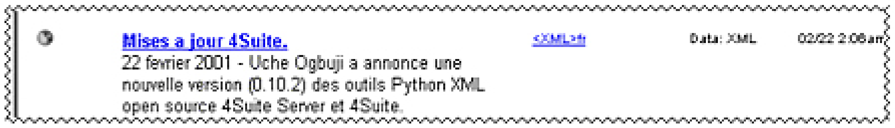
```
<p>
  Le <a href="http://4suite.org/index.epy">site</a>&#160;
  <object.title>4Suite</object.title> d&#233;crit <object.title>4Suite
  Server</object.title> comme un "une architecture pour services
  <object.title>XML</object.title>", n'&#233;tant pas destin&#233;e
&#224;
. &#234;tre un serveur d'applications autonome mais plut&#244;t &#224;
. "coop&#233;rer &#233;troitement avec d'autres technologies de
serveurs
  d'applications".
</p>
```

However, the literal keywords cannot be used as unique identifiers by themselves. (A good example of this is the need to distinguish between the Apache organization and the Apache web server.) The RSS 1.0 taxonomy module was defined to fix this issue by replacing the words ordinarily used within dc:subject elements with unique identifiers (URIs).

The topic URIs are simply constructed by concatenation of a base URI, the element name, and the text content of the element. Here's an example of an item description using RSS 1.0 and the DC and taxonomy modules:

```
<item rdf:about="http://xmlfr.org/actualites/tech/010222-0001">
  <title>Mises &#224; jour 4Suite.</title>
  <link>http://xmlfr.org/actualites/tech/010222-0001</link>
  <dc:description>Uche Ogbuji a annonc&#233; une
.../...</dc:description>
  <dc:creator>Par Michael Smith, xmlhack - traduit par Eric
      van der Vlist, Dyomedea (vdv@dyomedea.com).</dc:creator>
  <dc:date>2001-02-22</dc:date>
  <dc:subject>4Suite Server, 4Suite, Uche Ogbuji, .../... </dc:subject>
  <taxo:topics>
    <rdf:Bag>
      <rdf:li
resource="http://xmlfr.org/index/object.title/4suite+server/"/>
      <rdf:li resource="http://xmlfr.org/index/object.title/4suite/"/>
      <rdf:li resource="http://xmlfr.org/index/person/uche+ogbuji/"/>
      <rdf:li resource="http://xmlfr.org/index/object.title/python/"/>
             .../...
    </rdf:Bag>
  </taxo:topics>
  <dc:publisher>XMLfr</dc:publisher>
  <dc:type>text</dc:type>
  <dc:language>fr</dc:language>
</item>
```

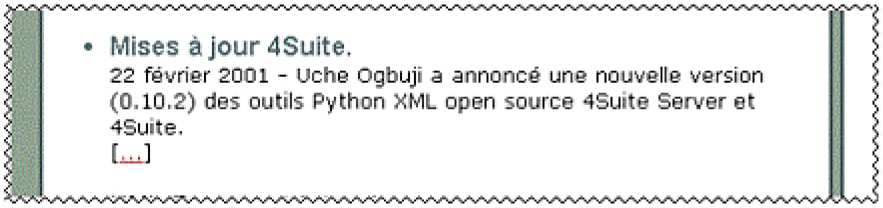
This is fun, but do we have a use for such a document? The basic use of RSS channels today is to get the titles of stories on your site displayed by aggregators such as O'Reilly's Meerkat.



Mises a jour 4Suite. <XML>fr Data: XML 02/22 2:08am
22 fevrier 2001 - Uche Ogbuji a annonce une nouvelle version (0.10.2) des outils Python XML open source 4Suite Server et 4Suite.

.

And in our case, XMLfr uses these channels internally to display its lists of articles, giving RSS its original meaning of "RDF Site Summary":



- Mises à jour 4Suite.
  22 février 2001 - Uche Ogbuji a annoncé une nouvelle version (0.10.2) des outils Python XML open source 4Suite Server et 4Suite.
  [...]

Aggregators of RSS information (Meerkat, Morever, etc.) do not *yet* use taxonomy information, and a simple RSS channel would be sufficient to get your title displayed. So how can we utilize the extra metadata we extracted from the document body?

RDF Databases

One significant feature of RSS 1.0 is that it is fully compliant with RDF and can be directly loaded into RDF databases such as rdfDB or Squish. These let you query the data using an SQL-like query language and give you full access to the taxonomy information.

These languages are very convenient for walking through the entire set of RDF triples, letting you access all the information that is available by doing joins between related objects. The following example, using rdfDB, shows queries to find all the articles that mention *the person Uche Ogbuji* and then to display *all the topics from articles that mention the person Uche Ogbuji*.

```
load RDF file http://xmlfr.org/actualites/general.rss10 into newrss</>
0
0 </>
select ?item from newrss where
  (http://purl.org/rss/1.0/modules/taxonomy/#topics ?item  ?bag),
  (http://www.w3.org/1999/02/22-rdf-syntax-ns##li
    ?bag  http://xmlfr.org/index/person/uche+ogbuji/)
</>
?item
http://xmlfr.org/actualites/tech/010222-0001
0 </>
select ?topic from newrss where
 (http://www.w3.org/1999/02/22-rdf-syntax-ns##li
    ?bag  http://xmlfr.org/index/person/uche+ogbuji/)
 (http://www.w3.org/1999/02/22-rdf-syntax-ns##li ?bag ?topic)
```

```
</>
?topic
http://xmlfr.org/index/org/fourthought/
.../...
http://xmlfr.org/index/object.title/python/
http://xmlfr.org/index/person/uche+ogbuji/
http://xmlfr.org/index/object.title/4suite/
http://xmlfr.org/index/object.title/4suite+server/
0 </>
```

XMLfr has been running for several months using rdfDB as the backend storage for its dynamic index system and using JrdfDB, a Java interface developed for this purpose that interfaces with the XSLT processor XT.

Although rdfDB has been fast and reasonably stable, several features are badly needed for this application to be scalable and to develop additional features. These features include

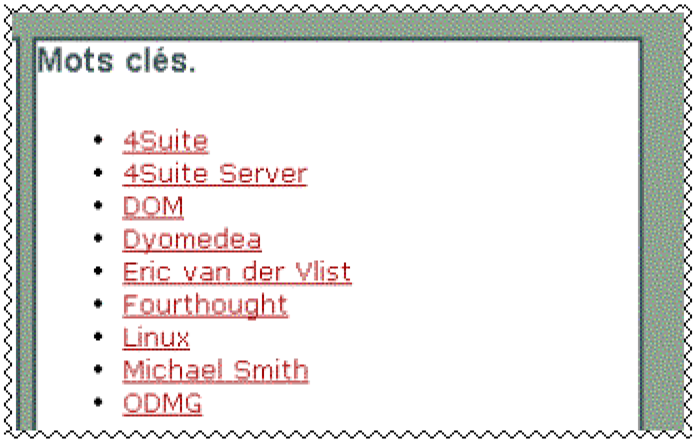
- Sorting (to retrieve articles by date)
- Retrieving unique rows (to remove duplicate results)
- Setting a maximum number of rows (to paginate the results)
- Grouping
- Aggregates (to count a number of relevant matches)
- Administration (load/unload)

RDBMS

So if rdfDB won't perform, what next? A fully fledged RDF database is not strictly needed just to keep track of the relations between topics and pages (or "occurrences," to follow the vocabulary of Topic Maps), and a traditional RDBMS with a straightforward table design has all the qualities required to be used as online storage for this purpose. XMLfr has migrated its dynamic index to a couple of PostgreSQL tables:

```
test=> \d topics
Table    = topics
+---------------------+----------------------------------+-------+
|       Field         |              Type                | Length|
+---------------------+----------------------------------+-------+
| channel             | varchar()                        |   255 |
| item                | varchar()                        |   255 |
| topic               | varchar()                        |   255 |
+---------------------+----------------------------------+-------+
test=> \d items
Table    = items
+----------------------+----------------------------------+--------+
|        Field         |              Type                | Length|
+----------------------+----------------------------------+--------+
| item                 | varchar()                        |   255 |
| dcdate               | date                             |     4 |
| title                | varchar()                        |   255 |
| description          | varchar()                        |   255 |
+----------------------+----------------------------------+--------+
```

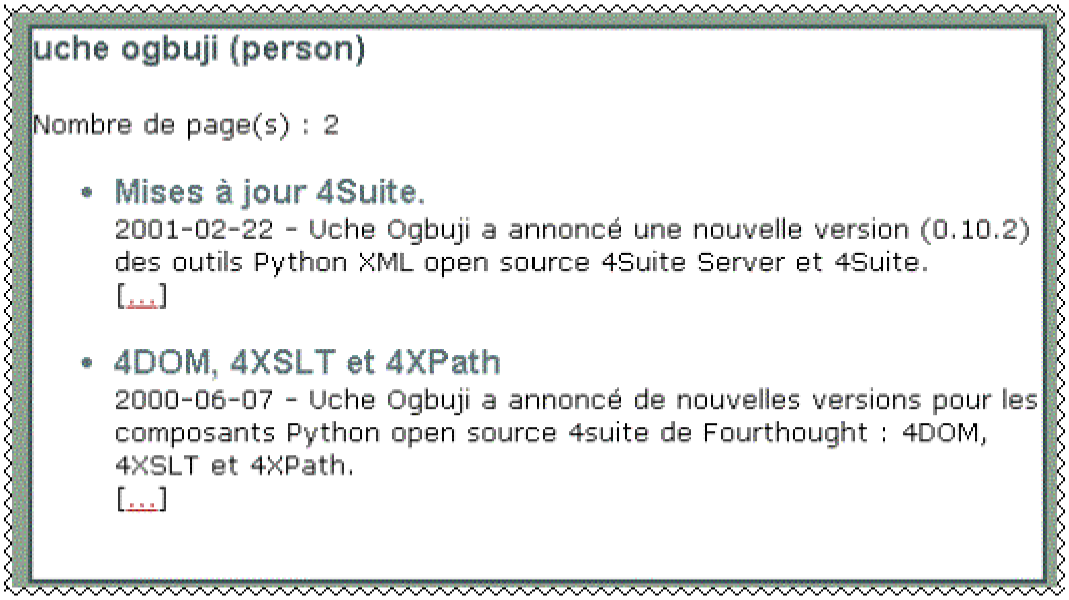
These tables are loaded with data from text dumps, which are generated by two simple XSLT transformations run against the RSS 1.0 channels. The dynamic index system on XMLfr is reached through a table of keywords displayed with the articles:



Mots clés.

- 4Suite
- 4Suite Server
- DOM
- Dyomedea
- Eric van der Vlist
- Fourthought
- Linux
- Michael Smith
- ODMG

These keywords are linked to pages from the dynamic index, displaying lists of articles found in the database:



uche ogbuji (person)

Nombre de page(s) : 2

- Mises à jour 4Suite.
  2001-02-22 - Uche Ogbuji a annoncé une nouvelle version (0.10.2) des outils Python XML open source 4Suite Server et 4Suite.
  [...]

- 4DOM, 4XSLT et 4XPath
  2000-06-07 - Uche Ogbuji a annoncé de nouvelles versions pour les composants Python open source 4suite de Fourthought : 4DOM, 4XSLT et 4XPath.
  [...]

These results are very similar to those we might obtain from the creation of a Topic Map.

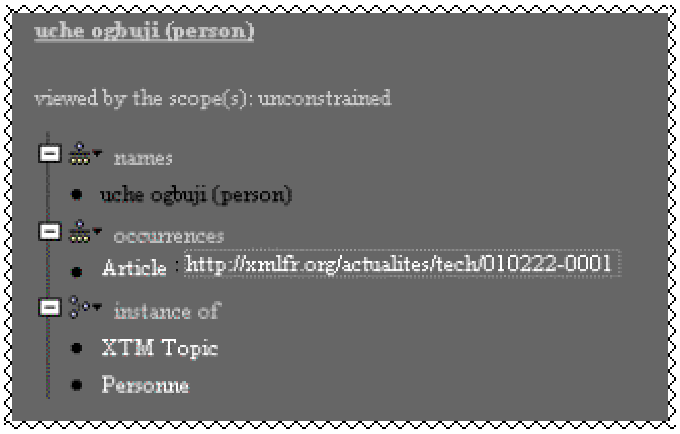
Topic Maps

Topic Maps are documents that describe topics, their interrelations, and their occurrences within an XML document.

A RSS 1.0 channel with taxonomy data happens to have all the information needed to generate a XTM 1.0 Topic Map, as the following example Topic Map fragment shows.

```
<topic id="person-uche+ogbuji">
  <instanceOf>
    <topicRef xlink:href="#person"/>
  </instanceOf>
  <baseName>
    <baseNameString>uche ogbuji (person)</baseNameString>
  </baseName>
  <occurrence id="person-uche+ogbuji-1">
    <instanceOf>
      <topicRef xlink:href="#story"/>
    </instanceOf>
    <resourceRef xlink:href="http://xmlfr.org/actualites/tech/010222-
0001"/>
  </occurrence>
</topic>
```

As a proof of concept, this Topic Map has been loaded into the empolis K42Â? Knowledge Server, a screenshot of which is shown below.


uche ogbuji (person)

viewed by the scope(s): unconstrained

names
- uche ogbuji (person)

occurrences
- Article : http://xmlfr.org/actualites/tech/010222-0001

instance of
- XTM Topic
- Personne

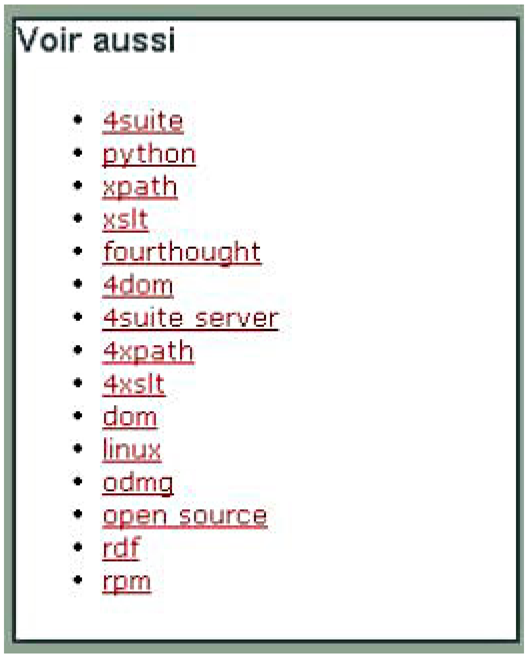
Although the screenshot looks a bit different from the list of articles on the web site, the difference doesn't add value and shows that the actual syntax doesn't matter -- the dynamic index is essentially a Topic Map. With some effort, and the features of an RDBMS, we can also do more than this by creating more information that describes how the topics over the site are related.

Aerial Photographs

A Topic Map of XMLfr maps the site content and gives the same picture -- in a different syntax -- as the dynamic index system available online.
This picture is directly derived from the markup used in the articles published on the site, and adding a new keyword in a story marked up as org, object.title, or person is sufficient to create a new topic.

The obvious things that are missing from this Topic Map is the topic associations, that is, the relationships between the topics in the map.

However, if we do not *a priori* know the nature of the topic associations, we may guess at their existence by looking at the most common associations found in the articles. This feature can easily be achieved using SQL grouping and aggregates; and it's been implemented on XMLfr through a very simple algorithm: for each topic, the list of the 15 other topics most often found associated with the current topic is displayed:



Voir aussi

- 4suite
- python
- xpath
- xslt
- fourthought
- 4dom
- 4suite server
- 4xpath
- 4xslt
- dom
- linux
- odmg
- open source
- rdf
- rpm

The accuracy of this technique in discovering related topics is surprising. As an example, Tim Berners-Lee is associated with XML, W3C, RDF, SVG, URI, W3C, XLink, DOM, HTML, HTTP, Java, SGML, Semantic Web, XPath and ISO, which is a fairly good description for such a simple algorithm.

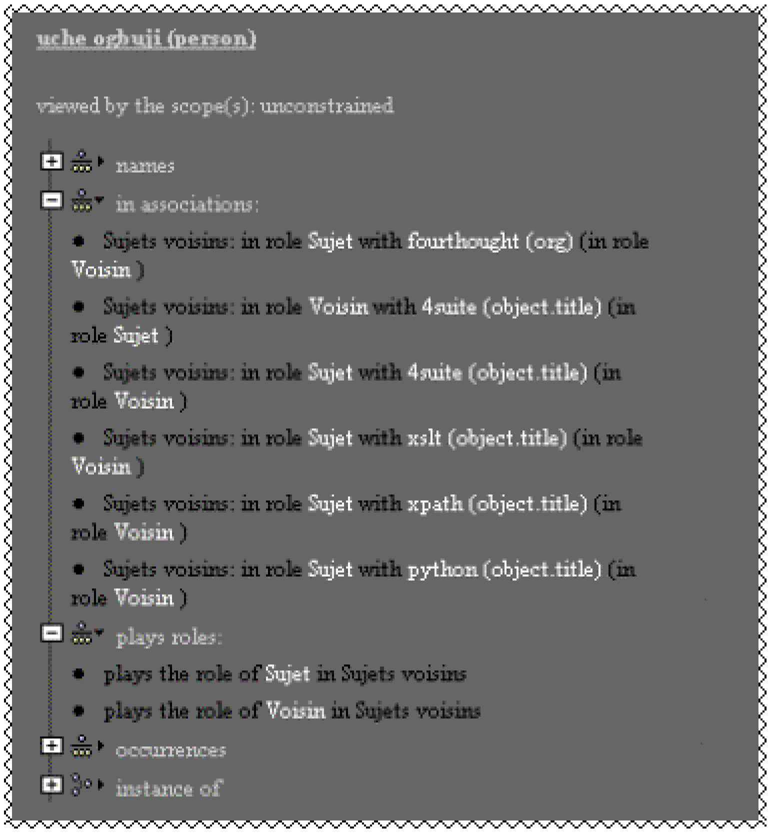
This same algorithm couldn't, unfortunately, be used directly on current RDF databases, as they are missing aggregates and grouping.

It can be used to generate associations in our XTM Topic Map, as shown by this fragment below which shows a relationship between the person *Uche Ogbuji*and the object *4Suite.*

```
      <association id="assoc-person-uche+ogbuji-2">
              <instanceOf>
                      <topicRef xlink:href="#related"/>
              </instanceOf>
              <member>
                      <roleSpec>
                              <topicRef xlink:href="#from"/>
                      </roleSpec>
                      <topicRef xlink:href="#person-uche+ogbuji"/>
              </member>
              <member>
                      <roleSpec>
                              <topicRef xlink:href="#to"/>
                      </roleSpec>
                      <topicRef xlink:href="#object.title-4suite"/>
              </member>
        </association>
```

These associations form patterns over the collection of articles, similar to the curves that can be seen on an aerial photograph: a human is needed to say if it's a road or a river and thus turn the photograph into a map. But I think the patterns should be usable as a first step for finding topic associations.

The associations have been created, in this Topic Map, as almost anonymous (related/from/to) and could be manually updated to transform the Topic Aerial Photograph into a Topic Map.



uche ogbuji (person)

viewed by the scope(s): unconstrained

names

in associations:

- Sujets voisins: in role Sujet with fourthought (org) (in role Voisin)
- Sujets voisins: in role Voisin with 4suite (object.title) (in role Sujet)
- Sujets voisins: in role Sujet with 4suite (object.title) (in role Voisin)
- Sujets voisins: in role Sujet with xslt (object.title) (in role Voisin)
- Sujets voisins: in role Sujet with xpath (object.title) (in role Voisin)
- Sujets voisins: in role Sujet with python (object.title) (in role Voisin)

plays roles:

- plays the role of Sujet in Sujets voisins
- plays the role of Voisin in Sujets voisins

occurrences

instance of

Tout ce Transforme

Antoine-Laurent de Lavoisier, a French chemist, once said, *Rien ne se perd, rien ne se crée, tout se transforme,*, or, "nothing is lost, nothing is created, everything is transformed"; and French people believe that this sentence is the foundation of modern chemistry. The real enabler for the work described in this article is of course XSLT, by which "everything istransformed".

Although, unlike Lavoisier's discovery, an XSLT transformation does allow the loss of content (this is sometimes referred as "semantic firewall"), an XSLT transformation *does not create anything*, so this result wouldn't have been possible if the source documents hadn't been carefully tagged.

This clearly shows that even if new technologies are now available to manipulate semantic information, this information needs to be available in the original documents, manually added afterward, or automatically extracted -- this is one of the challenges of Semantic Web.

Credits

Many thanks to

- The RSS 1.0 Working Group.
- Bénédicte Le Grand for a presentation at XML 2000 (Conceptual Exploration of Topic Maps) that gave me some hints on how to calculate the distance between topics.
- Empolis (previously known as Step UK) for kindly lending me a license of their k42 Topic Map Engine and patiently supporting my questions.
- The authors of the many open source software products used to run XMLfr (Linux, Apache, Jserv, XT, PostgreSQL, etc.).

References

- XMLfr: http://xmlfr.org/
- XMLNews-Story: http://www.xmlnews.org/docs/xmlnews-story.html
- RSS 1.0: http://purl.org/rss/1.0/
- RDF: http://www.w3.org/TR/REC-rdf-syntax/
- rdfDB: http://web1.guha.com/rdfdb/
- JrdfDB: http://4xt.org/downloads/JrdfDB/
- Squish: http://swordfish.rdfweb.org/rdfquery/
- XTM 1.0: http://www.topicmaps.org/xtm/1.0/
- Empolis k42: http://www.empolis.co.uk/products/prod_k42.asp
- Le Grand, Bénédicte - Conceptual Exploration of Topic Maps (XML 2000 conference presentation)